|  |  |
| --- | --- |
| Name: | Arpitha S P |
| Lab User ID: | 23SEK3324\_U10 |
| Date: | 09-01-2024 |
| Application Name: | Juice-Shop |

**Follow the below guidelines:**





System Architecture:

(Understand the system and document the physical and logical architecture of the system, use the shapes and icons to capture the system architecture)

EC2 Instance-VM

Docker Engine

Container

13.232.110.42:3000

Docker Image

Physical Architecture :

* EC2 Instance
  + Instance type : t2 medium
  + Platform : ubuntu (Linux/UNIX)
  + Volume size(GiB) : 20
  + Availability zone : ap-south-1a

Logical Architecture :

* Docker Containers : These are isolated instances running within docker engine
* Docker Engine manages and orchestrates these containers

Define system’s normal behavior:

(Define the steady state of the system is defined, thereby defining some measurable outputs which can indicate the system’s normal behavior)

1. Availability : The application is accessible and available to users without downtime or disruptions.

* Users can navigate the site, view products and place orders without encountering errors or unresponsive pages.

1. Performance : Pages load reasonably fast, transactions complete without delays, and user interactions remain responsive.
2. Security : The application is secure, free from known vulnerabilities, and follows best practice to protect user data.
3. Functionality : Users can browse products , add items to carts, complete purchases, and interact with various elements without encountering bugs.
4. Monitoring : Continuous monitoring to identify potential issues or threats.

Hypothesis:

(During an experiment, we need a hypothesis for comparing to a stable control group, and the same applies here too. If there is a reasonable expectation for a particular action according to which we will change the steady state of a system, then the first thing to do is to fix the system so that we accommodate for the action that will potentially have that effect on the system. For eg: "If one of our database servers fails, our service will automatically switch to a backup server, and users will not experience any downtime or data loss.")



* Current Docker Configuration
* Historical system stability data
* Optimal configurations for resource
* Impact of docker on system reliability

**Known**

Things we are aware of but don’t understand.

Things we are aware of and understand.

* Compatibility with future Docker Versions
* Employee adaption to docker usage

**Unknown**

* Unforeseen bugs in Docker configurations
* Unpredicted events affecting stability.

**Unknown**

**Known**

Things we are neither aware of nor understand.

Things we understand but are not aware of.

**Known-Knowns :**

* Docker Configuration: This refers to the current setup and configuration of Docker that is well-understood. It includes details like the docker version in use, the existing container configurations etc.,.
* System stability data: Historical data on system stability is a known known. It provides insights into the system’s performance over time.

**Known-Unknowns:**

* Optimal Configurations for Resource Usage: There may be uncertainties regarding the best configurations for resource usage. This could include determining the optimal allocation of CPU, Memory and other resources for containers to maintain system stability and performance.
* Impact of Docker on System Reliability: While the historical stability data is known, the specific impact of introducing Docker on a system reliability may be uncertain.

**Unknown-Knowns:**

* Compatibility with future Docker Versions: This involves aspects that are available but not fully recognized or understood. It might relate to how well the current docker configurations align with potential updates or changes in future Docker versions
* Employee Adaptation to Docker Usage: This involves factors that are available but might not be fully recognized, such as how well employees adapt to the use of Docker.

**Unknown-Unknowns:**

* Unforeseen Bugs in Docker Configurations: These are unexpected issues that may arise during the use of Docker, potentially due to bugs in the configurations. Unanticipated events, security vulnerabilities, or compatibility issues that weren’t initially known.
* Unpredicted Events affecting stability: These are unforeseen events that could impact system stability and might not have been considered initially. It could include factors like sudden changes in user behavior, unexpected traffic spikes, or external events affecting the application.

Experiment:

(Document your Preparation, Implementation, Observation and Analysis )

Juice Shop is written in Node.js, Express and Angular. It was the first application written entirely in JavaScript listed

in the OWASP VWA Directory.

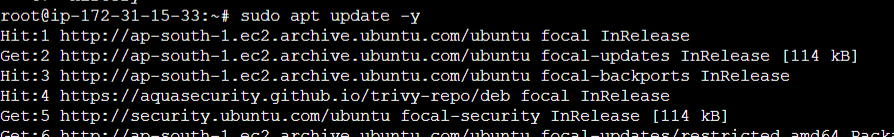
The OWASP Juice Shop has been created by Björn Kimminich and is developed, maintained and translated by a team

of volunteers.

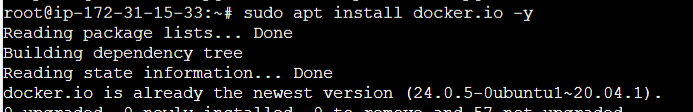
This application consists of a various levels of hacking challenges where the user is supposed to exploit the underlying

vulnerabilities.

Step 1 : Updating the packages

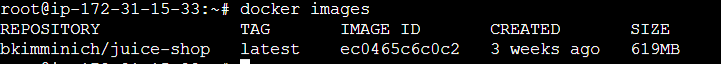


Step 2 : Installing Docker



Step 3 : Pull image

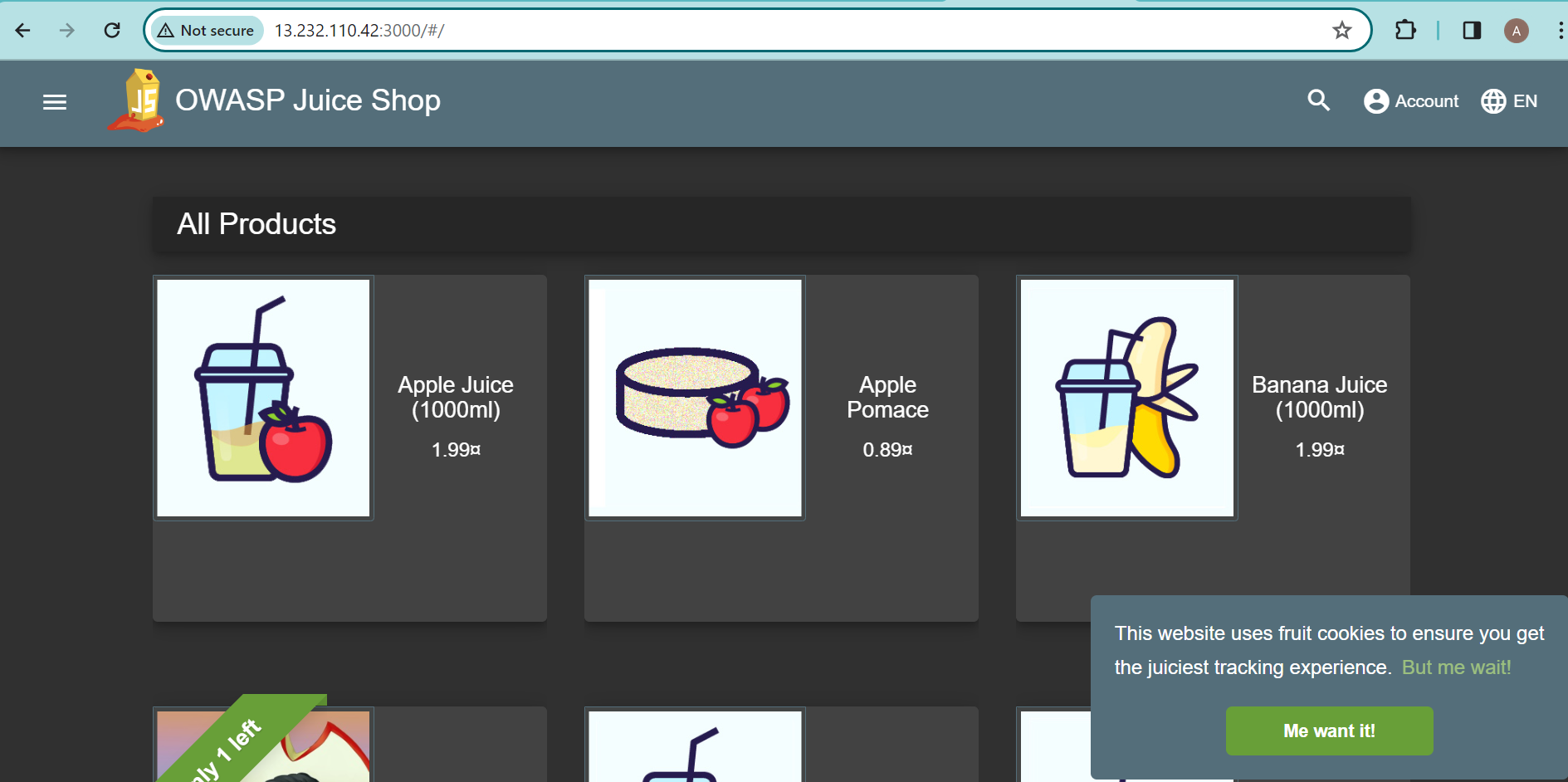




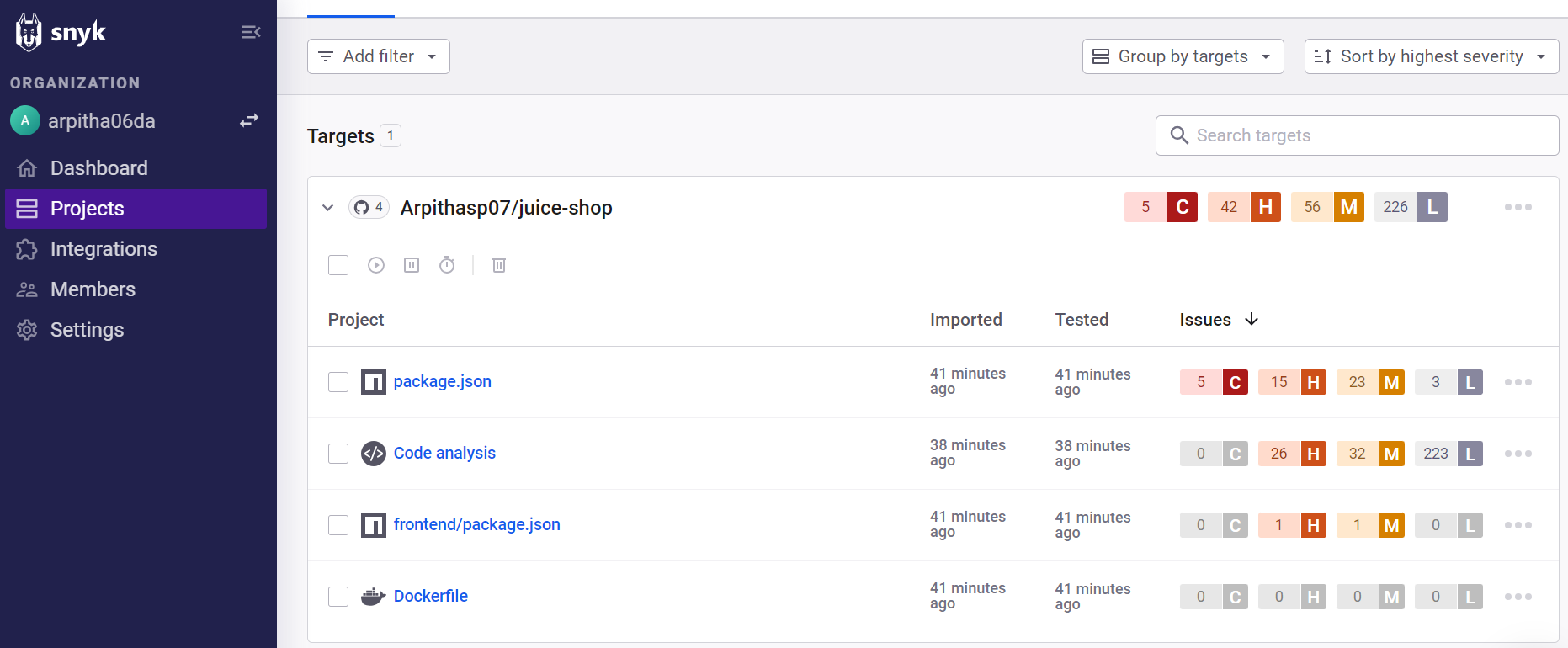
Step 4 : Docker Run







**Vulnerability Check using snyk:**



**Docker :**

Docker is a platform designed to make it easier to create, deploy, and run applications using containers. It provides

tools and platform to manage these containers, making it simpler to build and run applications consistently across

different environments.

**Docker pull** : It is a command used to download Docker images from a container registry, like docker hub to local

machine .

**docker pull bkimminich/juice-shop** – It fetches the juice-shop image and stores it in the system.

Docker pull enables us to obtain a copy of the desired Docker image, making it available for use in creating containers

or running applications.

**Docker run :** docker run is used to create and start a container based on a specific docker image.

docker run --rm -p 3000:3000 bkimminich/juice-shop – docker run initiates the creation and launch of the container.

**‘—rm’** : removes the container automatically after it stops running. Keeps your system clean from unused containers

**‘-p 3000:3000’** : port mapping . It maps port 3000 of the host machine to port 3000 of the container which allows us

to access the application running inside the container via port 3000 on your local machine.

**Snyk :**

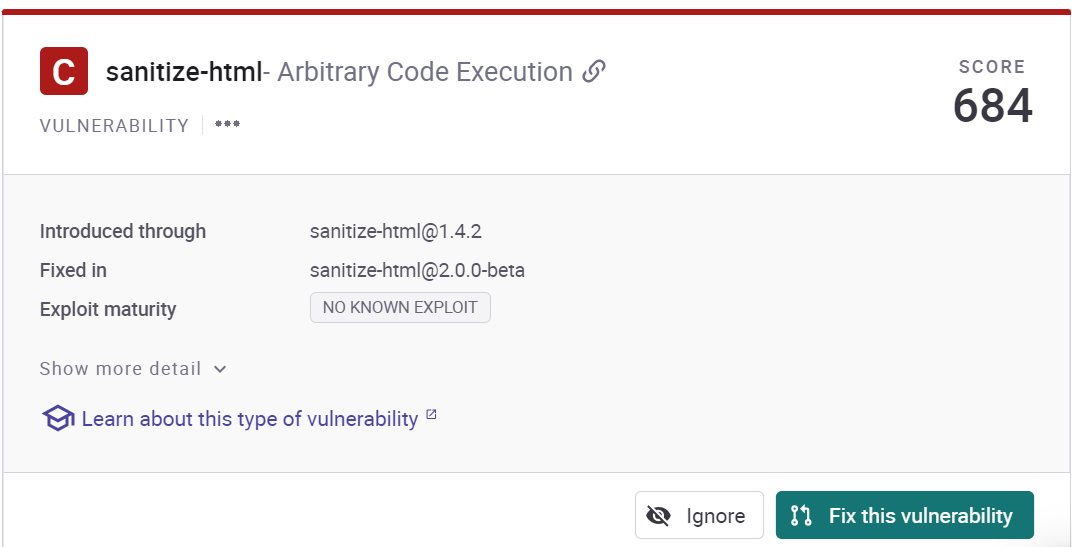
Snyk is a security tool used to help developers identify, fix, and prevent vulnerabilities in their applications.It scans

container images for known vulnerabilities and security issues.

**Observation :**

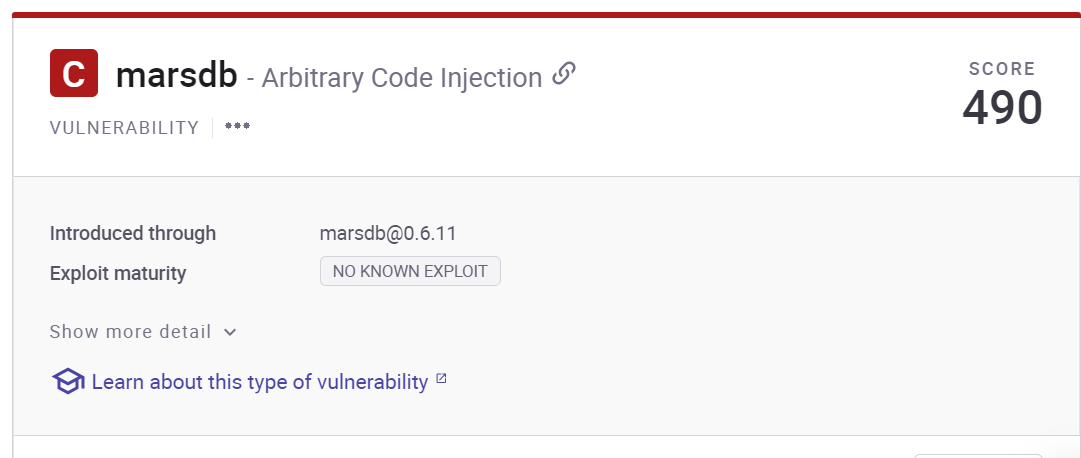
There are 5 CRITICAL , 42 HIGH , 56 MEDIUM, 226 LOW vulnerabilities.

CRITICAL :



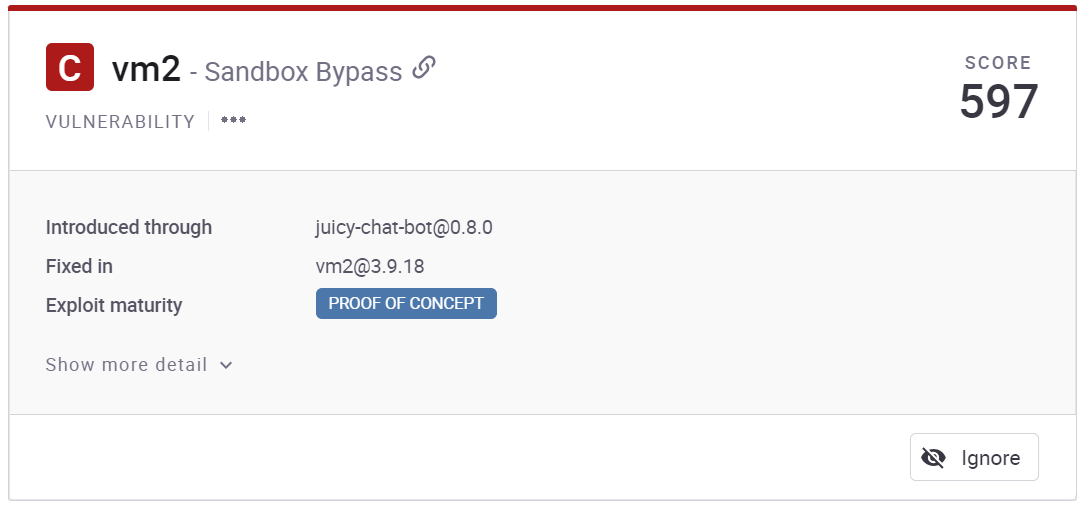
sanitize-html is a library that allows you to clean up user-submitted HTML, preserving whitelisted elements and whitelisted attributes on a per-element basis. Affected versions of this package are vulnerable to Arbitrary Code Execution. Tag transformations which turn an attribute value into a text node using transformTags could be vulnerable to code execution.

Solution : Upgrade sanitize-html to version 2.0.0-beta or higher.



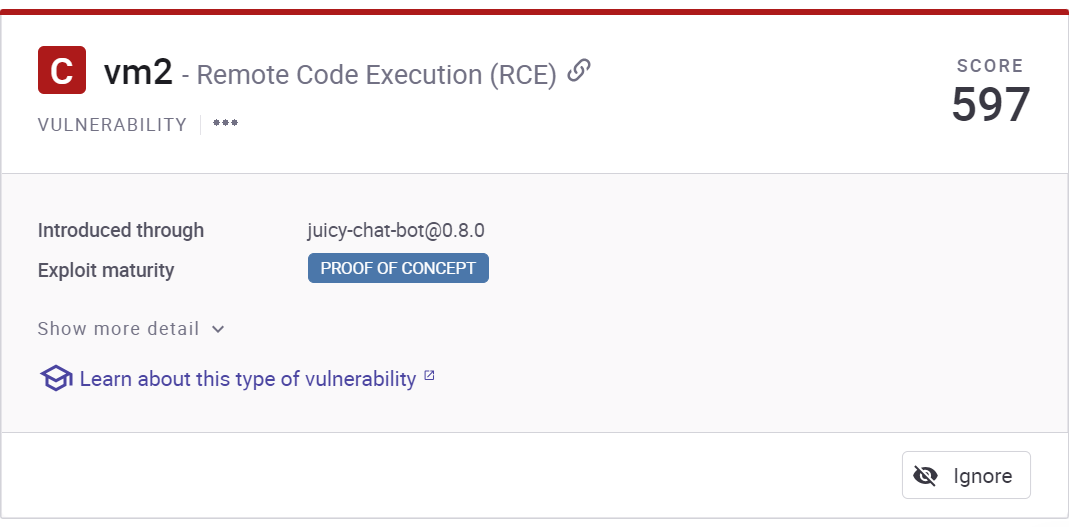
marsdb is a MarsDB is a lightweight client-side database. Affected versions of this package are vulnerable to Arbitrary Code Injection. In the DocumentMatcher class, selectors on $where clauses are passed to a Function constructor unsanitized. This allows attackers to run arbitrary commands in the system when the function is executed.

Solution : No remediation path available.



vm2 is a sandbox that can run untrusted code with whitelisted Node's built-in modules. Affected versions of this package are vulnerable to Sandbox Bypass by abusing an unexpected creation of a host object based on the maliciously crafted specification of Proxy. Exploiting this vulnerability allows an attacker to gain remote code execution rights on the host running the sandbox via the Function constructor.

Solution : No remediation path available.



vm2 is a sandbox that can run untrusted code with whitelisted Node's built-in modules. Affected versions of this package are vulnerable to Remote Code Execution (RCE) such that the Promise handler sanitization can be bypassed, allowing attackers to escape the sandbox.

Solution : No remediation path available.